Bulletin of the *Transilvania* University of Braşov • Vol 7(56), No. 2 - 2014
Series III: Mathematics, Informatics, Physics, 169-176

MINIMUM FLOWS IN NETWORKS WITH PARAMETRIC LOWER BOUNDS FOR SINK ARCS

Eleonor CIUREA[1]

Abstract

In this paper, we study minimum flows in network with parametric lower bounds for sink arcs problem. In this problem, the lower bounds for sink arcs are a nonincreasing linear function of a parameter λ, and we wish to determine a minimum flow for q values $0 = \lambda_1, \lambda_2, \ldots, \lambda_q$ of parameter λ. Assume, that $0 = \lambda_1 < \lambda_2 < \ldots < \lambda_q$ and $q \leq n$, where n is a number of nodes.

2000 *Mathematics Subject Classification:* 90B10, 90C35, 05C35, 68R10.

Key words: network flow, networks algorithms, minimum flow problem, paramatric networks.

1 Minimum flow problem

The theory of flow is one of the most important parts of Combinatorial Optimization. The maximum flow problem is presented in books [1], [2], [5]. In this section we discuss some basic notation and results for minimum flow problem, which are presented in paper. [3].

Let $\mathbb{N}$ be the natural number set and $G = (N, A, l, u)$ a network with the nodes set $N = \{1, \ldots, n\}$, the arcs set $A = \{a_1, \ldots, a_k, \ldots, a_m\}$, $a_k = (i, j)$, the lower bound function $l : A \to \mathbb{N}$, the upper bound (capacity) function $u : A \to \mathbb{N}$, 1 the source node and n the sink node.

For a given pair of subset X, Y of the nodes set N we use the notation: $(X, Y) = \{(i, j) | (i, j) \in A, i \in X, j \in Y\}$ and for a given function $g : A \to \mathbb{N}$ we use the notation

$$g(X, Y) = \sum_{(X,Y)} g(i, j)$$

A flow is a function $f : A \to \mathbb{N}$ satisfying the next conditions:

[1]Faculty of Mathematics and Informatics, *Transilvania* University of Braşov, Romania, e-mail: e.ciurea@unitbv.ro

$$f(i,N)-f(N,i)=\begin{cases} v, & \text{if } i=1 \\ 0, & \text{if } i\neq 1,n \\ -v, & \text{if } i=n \end{cases} \tag{1.1}$$

for some $v \geq 0$. We refer to v as the value of flow f. A flow is called feasible if f satisfies the following constraints:

$$l(i,j) \leq f(i,j) \leq u(i,j), (i,j) \in A, \tag{1.2}$$

The minimum flow problem is to determine a flow f for which v is minimized.

A preflow f is a function $f : A \to \mathbb{N}$ satisfying the next conditions:

$$f(i,N)-f(N,i) \leq 0, i \in N-\{1,n\} \tag{2.1}$$
$$l(i,j) \leq f(i,j) \leq u(i,j), (i,j) \in A \tag{2.2}$$

For any preflow f, we define the deficit of node i as

$$e(i)=f(i,N)-f(N,i), i \in N \tag{3}$$

We refer to a node i with $e(i)=0$ as balanced. A preflow f satisfying the condition $e(i)=0, i \in N-\{1,n\}$ is a flow. Thus, a flow is a particular case of preflow.

We further assume, without loss of generality, that if $(i,j) \in A$ the $(j,i) \in A$ (if $(j,i) \notin A$ we consider that $(j,i) \in A$ with $l(i,j)=u(i,j)=0$).

A cut is a partition of the nodes set N into two subsets X and $\bar{X}=N-X$. We represent this cut using the notation $[X,\bar{X}]$. We refer to a cut $[X,\bar{X}]$ as a $1-n$ cut if $1 \in X$ and $n \in \bar{X}$. An arc (i,j) with $i \in X$ and $j \in \bar{X}$ is a forward arc of the cut, and an arc (i,j) with $i \in \bar{X}$ and $j \in X$ is a backward arc of the cut. Let $(X,\bar{X})$ denote the set of backward arcs, and let $(\bar{X},X)$ denote the set of backwards arcs. Hence, all the arcs of a $1-n$ cut are $[X,\bar{X}]=(X,\bar{X})\bigcup(\bar{X},X)$.

For the minimum flow problem, the capacity $c[X,\bar{X}]$ of a $1-n$ cut is $c[X,\bar{X}]=l(X,\bar{X})-u(\bar{X},X)$. We refer to a $1-n$ cut whose capacity is the maximum among all $1-n$ cuts as a maximum cut.

Theorem 1. *The value of the minimum flow from a source node* 1 *to a sink node* n *in a network* $G=(N,A,l,u)$ *equals the capacity of the maximum* $1-n$ *cut.*

For the minimum flow problem, the residual capacity $\hat{r}(i,j)$ of any arc $(i,j) \in A$, with respect to a given preflow f is given by $\hat{r}(i,j)=c(j,i)-f(j,i)+f(i,j)-l(i,j)$. The residual network is $\hat{G}=(N,\hat{A},\hat{r})$, $\hat{A}=\{(i,j)|(i,j) \in A, \hat{r}(i,j)>0\}$.

The minimum flow problem in a network $G=(N,A,l,u)$ can be solved in two phases:

(1) establishing a feasible flow;

(2) from a given feasible flow, establish a minimum flow.

The solution of phase one is presented in [1], [3], [5]. There are three approaches for solving the minimum flow problem: (1) using decreasing path algorithms, (2) using preflow algorithms and (3) minmax algorithms. These approaches are presented in [3]. The preflow algorithms are the following: generic preflow algorithm, FIFO preflow algorithm, highest label preflow algorithm and deficit scaling algorithm. Because in Section 2 we use a variant of FIFO preflow algorithm, next we present this algorithm.

Before describing the FIFO preflow algorithm for the minimum flow problem, we introduce some definitions. In the residual network $\hat{G}$, the distance function d is a function $d : N \to \mathbb{N}$. We say that a distance function is valid if it satisfies the following conditions: $d(1) = 0, d(j) \leq d(i) + 1, (i, j) \in \hat{A}$. We refer to $d(i)$ as the distance label of node i. We say that the distance labels are exact if for each node i, $d(i)$ equals the length of the shortest path from source node 1 to node i in the residual network $\hat{G} = (N, \hat{A}, \hat{r})$. We refer to an arc $(i, j) \in \hat{A}$ as an admissible arc if $d(j) = d(i) + 1$; otherwise it is inadmissible. We refer to a path from source node 1 to sink node n, in residual network $\hat{G}$, consisting entirely of admissible arcs as an admissible path; otherwise it is inadmissible. We denote by $\hat{A}^-(j)$ the arcs set $\hat{A}^-(j) = \{(i, j)|(i, j) \in \hat{A}\}$. We assume $\hat{A}^-(j)$ to have a fixed order. For each node j, there always is a distinguished current arc. Initially, this arc is always the first arc of $\hat{A}^-(j)$.

The FIFO preflow algorithm for the minimum flow problem is presented in Figure 1.

```
1: FIFO PF;
2: BEGIN
3: PREPROCESS;
4: while L ≠ 0 do
5:     BEGIN
6:     remove the j from the front of the queue L;
7:     PULL/RELABEL(j);
8:     END
9: end while
10: END.
```

```
1: PROCEDURE PREPROCESS;
2: BEGIN
3: let f be a feasible flow in network G;
4: compute the residual network Ĝ;
5: compute the exact distance labels d(•) in Ĝ;
6: if d(t) ≥ n then
7:     f is a minimum flow;
8: else
9:     BEGIN
10:    L := ∅;
```

```
11:     for (i, n) ∈ Â⁻(n) do
12:         BEGIN
13:         f(i, n) := r̂(i, n);
14:         if (e(i) < 0) AND (i ≠ 1) then
15:             add i to the rear of L;
16:         end if
17:     end for
18:     d(n) := n;
19: end if
20: END;
```

```
 1: PROCEDURE PULL/RELABEL(j);
 2: BEGIN
 3: select the first arc (i, j) ∈ Â⁻(j);
 4: b := 1;
 5: repeat
 6:     if (i, j) is an admissible arc then
 7:         BEGIN
 8:         pull g := min{−e(j), r̂(i, j)} units of flow from node j to node i;
 9:         if (i ∉ L) AND (i ≠ 1) AND (i ≠ t) then
10:             add i to the rear of L;
11:         end if
12:         END
13:     end if
14:     if e(j) < 0 then
15:         if (i, j) is not the last arc in Â⁻(j) then
16:             select the next arc in Â⁻(j) as current arc
17:         else
18:             BEGIN
19:             d(j) := min{d(i) + 1|(i, j) ∈ Â};
20:             b := 0;
21:             END
22:         end if
23:     end if
24: until (e(j) = 0) OR (b = 0);
25: if e(j) < 0 then
26:     add j to the rear of L;
27: end if
28: END;
```

Figure 1: The FIFO preflow algorithm

The FIFO preflow algorithm for minimum flow examines active nodes in the first in, first out (FIFO) order. The algorithm maintains the list L of the active nodes as a queue. A pull of g units of flow from node j to node i increases both $e(j)$

and $\hat{r}(j,i)$ by g units and decreases both $e(i)$ and $r(i,j)$ by g units. We refer to the process of increasing the distance label of node j, $d(j) := min\{d(i)+1|(i,j) \in \hat{A}\}$, as a relabel operation.

In paper [3] the follows two theorems are presented.

Theorem 2. *The FIFO preflow algorithm computes correctly a minimum flow.*

Theorem 3. *The FIFO preflow algorithm runs in $O(n^3)$ time.*

The proof of Theorem 3 uses the following remarks:

1) It partitions the total number of node examination into different phases. The first phase consists of node examinations for those nodes that become active during the procedure PREPROCESS. The second phase consist of the node examinations of all the nodes that are in the queue L after the algorithm has examined the nodes in the first phase and so on.

2) The total number of relabel operations is at most $2n^2$.

3) The total number of phases is at most $2n^2 + n$.

4) A bound of $O(nm)$ on the number of saturating pulls.

5) A bound of $O(n^3)$ on the number of nonsaturating pulls.

6) The bottleneck operation in the FIFO preflow algorithm is the number of nonsaturating pulls.

2 Minimum flows in networks with parametric lower bounds for sink arcs

Gallo, Grigoriadis and Tarjan [4] solve the source parametric maximum flow problem. In this problem, the upper bound of every source arc $(1,j)$ is a nondecreasing linear function of parameter λ, i.e. $\bar{u}(1,j) = u(1,j) + \lambda u_0(1,j)$ for some constant $u_0(1,j) \geq 0$. The upper bound of every other arc is fixed and they determine a maximum flow for q values $\lambda_1, \lambda_2, \ldots, \lambda_q$ with $0 = \lambda_1 < \lambda_2 < \ldots < \lambda_q$ and $q \leq n$.

In this Section we study the sink parametric minimum flow problem. In this problem, the lower bound of every sink arc (i,n) is a nonincreasing linear function of a parameter λ, i.e. $\bar{l}(i,n) = l(i,n) - \lambda l_0(i,n)$ for some constant $l_0(i,n) \geq 0$. The lower bound of every other arc is fixed. We assume that $\lambda \in \Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ with $0 = \lambda_1 < \lambda_2 < \ldots < \lambda_q$ and $q \leq n$. Let $\mathring{l}(i,n)$ be $\mathring{l}(i,n) = min\{l(i,n)|(i,n) \in A\}$ and let $\overset{*}{l}_0(i,n)$ be $\overset{*}{l}_0(i,n) = max\{l_0(i,n)|(i,n) \in A\}$. We assume that $\overset{*}{l}_0(i,n) \leq \mathring{l}(i,n)/\lambda_q$. In this case we have $\bar{l}(i,n) \geq 0$ for all $(i,n) \in A$.

Let MFP_k denote the minimum flow problem for specific value of $\lambda = \lambda_k$. Let v_k denote the minimum flow value of MFP_k and let $[X_k, \bar{X}_k]$ denote an associated maximum cut. Given a minimum flow f_k of MFP_k, we solve MFP_{k+1} as follows: with f_k as starting flow and the corresponding distance labels as the initial distance labels, we perform the procedure PREPROCESS by pulling additional flow along the sink arcs so that they all become saturated, i.e. $f_k(i,n) -$

$f'_{k+1}(i,n) = \bar{l}_{k+1}(i,n)$ with $\bar{l}_{k+1}(i,n) = l(i,n) - \lambda_{k+1} l_0(i,n)$. Then we apply the FIFO preflow algorithm.

The parametric FIFO preflow algorithm for solving all problems MFP_1, MFP_2, ..., MFP_q is presented in Figure 2.

```
1: PFIFO PF;
2: BEGIN
3: for k := 1 to q do
4:     FIFO PF;
5: end for
```

Figure 2: The parametric FIFO preflow algorithm

The procedure FIFO PF is the FIFO preflow algorithm presented in Figure 1 in which the feasible flow in network $G_k = (N, A, l_k, u)$ is the minimum flow f_{k-1} with f_0 the initial feasible flow.

Theorem 4. *The PFIFO PF algorithm computes correctly a minimum flow in network with parametric lower bounds for sink arcs.*

Proof. Results from Theorem 2 and the structure of the PFIFO PF. □

Theorem 5. *The ending distance labels of MFP_k are valid distances for MFP_{k+1} in the residual network $\hat{G}_k$ after the procedure PREPROCESS.*

Proof. The MFP_k and MFP_{k+1} are the same except that the lower bound of some sink arcs in MFP_{k+1} is less. So we take the minimum flow for the MFP_k and decrease the lower bound of the sink arcs, then all distance labels other than for the sink node satisfy the validity conditions. But when we perform the procedure PREPROCESS and saturate the sink arcs with positive residual capacities, then all distance labels resatisfy the validity conditions. Observe that in the procedure PREPROCESS we need to saturate only those arcs (i,n) for which $\hat{r}(i,n) > 0$ and $d(i) < n$. □

Theorem 6. *The PFIFO PF algorithm runs in $O(n^3)$ time.*

Proof. As all distance labels remain valid throughout, the total number of relabel in $O(n^2)$. This immediately implies that the arc saturations are $O(nm)$ and nonsaturating pulls are $O(n^3)$. Conclude that the PFIFO PF algorithm runs in $O(n^3)$ time. □

In the proof of Theorem 6 we taken into account the remarks presented in Theorem 3.

Theorem 7. *There are the inequalities $v_1 \geq v_2, \geq, \ldots, \geq v_q$ and some associated maximum cuts satisfy the nesting condition $\bar{X}_1 \subseteq \bar{X}_2 \subseteq \ldots \subseteq \bar{X}_q$.*

Proof. Obviously, $v_1 \geq v_2, \geq, \ldots, \geq v_q$. Let $\bar{X}_k$ define a maximum cut for MFP_k satisfying that each node $j \in \bar{X}_k$ is reachable from node n in the residual network $\hat{G}_k$. If this maximum cut contains no sink arc (i, n) with $l_0(i, n) > 0$, then the capacity of this cut does not increase by increasing λ and $\bar{X}_k$ also defines a maximum cut for $MFP_{k+1}, MFP_{k+2}, \ldots, MFP_q$ and the desired property holds. In case, the maximum cut defined by $\bar{X}_k$ contains a sink arc (i, n) with $l_0(i, n) > 0$, then the node $i \in X_k$ is reachable from sink node n in MFP_{k+1}. As all other nodes in $\bar{X}_k$ remain reachable from node n. It follows that $\bar{X}_k \bigcup \{i\} \subseteq \bar{X}_k + 1$ and alternatively, $\bar{X}_k \subseteq \bar{X}_{k+1}$. As this result is true for all k, we obtain $\bar{X}_1 \subseteq \bar{X}_2 \subseteq \ldots \subseteq \bar{X}_q$. □

References

[1] Ahuja, R. K., Magnanti, T. L. and Orlin, J. B. *Network flows. Theory, algorithms and applications*, Prentice Hall, Englewood, Cliffs, N. J., 1993.

[2] Bazaraa, M. S., Jarvis, J. J. and Sherali, H. D. *Linear programming and networks flows (third edition)*, Wiley, New York, 2005.

[3] Ciurea, E. and Ciupala, L., *Sequential and parallel algorithms for minimum flows*, Journal of Applied Mathematics and Computing, **15**, no. 1-2, (2004), 53-75.

[4] Gallo, G., Grigoriades, M. D. and Tarjan, R. E. *A fast parametric maximum flow algorithm and applications*, SIAM J. Comp., **18** (1989), 30-55.

[5] Jungnickel, D., *Graphs, networks and algorithms*, Springer, Berlin, 1999.